In [1]:

```
from keras.models import Sequential
from keras.layers import LSTM, Input, Dense, GRU, Embedding, Dropout
from keras.optimizers import RMSprop
from keras import activations
from keras.callbacks import EarlyStopping, ModelCheckpoint, TensorBoard, ReduceLROnPlateau
#from keras.initializers import RandomUniform
#from keras.initializers import Initializer

import numpy as np
import operator
import seaborn as sns
import matplotlib.pyplot as plt
import tensorflow as tf
import math




sns.set_style("whitegrid")
current_palette = sns.color_palette('colorblind')
```

```
Using TensorFlow backend.
```

In [2]:

```
features = 20 #entspricht der Anzahl der Sensoren
timesteps = 22 # *0.05s --> definiert die Zeitspanne in der zeitliche Abhängigkeiten vom Netzwerk er
batchsize = 128
LSTM_size = 64 #ANzahl der LSTM-Zellen
Dense_size = 32
epochen = 100


name = 'NN2_3_1_2_Mz'

#init = RandomUniform(minval=-0.05, maxval=0.05)
```

In [3]:

```
from keras.models import load_model

model = load_model('model/'+name)
```

Aufbau Model

In [76]:

```
model = Sequential()
model.add(LSTM(LSTM_size, dropout=0.3, recurrent_dropout=0.3 ,return_sequences=True,
               batch_input_shape=(None, timesteps, features)))
#model.add(LSTM(LSTM_size, dropout=0.3, recurrent_dropout=0.3 ,return_sequences=True))
model.add(LSTM(LSTM_size, dropout=0.3, recurrent_dropout=0.3 ,return_sequences=False))
#model.add(Dense(Dense_size))
```

```
model.add(Dense(Dense_size))
model.add(Dense(1, activation='linear'))

model.compile(loss='mse', optimizer='rmsprop')
```

Trainingsdaten laden

In [5]:

```
x_train = np.load('Regression_Daten/x_train.npy').astype('float32')
x_val = np.load('Regression_Daten/x_val.npy').astype('float32')
x_test = np.load('Regression_Daten/x_test.npy').astype('float32')

y_train = np.load('Regression_Daten/y_Mz_train2.npy').astype('float32')
y_val = np.load('Regression_Daten/y_Mz_val2.npy').astype('float32')
y_test = np.load('Regression_Daten/y_Mz_test2.npy').astype('float32')
```

Model trainieren

In [78]:

```
model.fit(x_train, y_train,
          batch_size=batchsize , epochs=epochen, validation_data=(x_val, y_val))
```

```
Train on 11392 samples, validate on 2944 samples
Epoch 1/100
11392/11392 [==============================] - 5s 435us/step - loss: 80.5287 - val_loss:
135.1637
Epoch 2/100
11392/11392 [==============================] - 3s 282us/step - loss: 78.8593 - val_loss:
134.9498
Epoch 3/100
11392/11392 [==============================] - 3s 280us/step - loss: 78.3108 - val_loss:
135.9605
Epoch 4/100
11392/11392 [==============================] - 3s 273us/step - loss: 77.5553 - val_loss:
135.4722
Epoch 5/100
11392/11392 [==============================] - 3s 282us/step - loss: 77.1010 - val_loss:
136.1593
Epoch 6/100
11392/11392 [==============================] - 3s 297us/step - loss: 77.2316 - val_loss:
138.2726
Epoch 7/100
11392/11392 [==============================] - 3s 281us/step - loss: 76.9244 - val_loss:
137.4398
Epoch 8/100
11392/11392 [==============================] - 3s 279us/step - loss: 77.0596 - val_loss:
137.2841
Epoch 9/100
11392/11392 [==============================] - 3s 295us/step - loss: 76.6431 - val_loss:
136.7351
Epoch 10/100
11392/11392 [==============================] - 3s 277us/step - loss: 76.3273 - val_loss:
136.3812
```

```
Epoch 11/100
11392/11392 [==============================] - 3s 291us/step - loss: 76.3023 - val_loss:
137.7915
Epoch 12/100
11392/11392 [==============================] - 3s 291us/step - loss: 75.9107 - val_loss:
137.1994
Epoch 13/100
11392/11392 [==============================] - 4s 308us/step - loss: 75.9203 - val_loss:
136.2373
Epoch 14/100
11392/11392 [==============================] - 4s 337us/step - loss: 76.1065 - val_loss:
135.6979
Epoch 15/100
11392/11392 [==============================] - 3s 307us/step - loss: 75.8515 - val_loss:
137.3448
Epoch 16/100
11392/11392 [==============================] - 4s 308us/step - loss: 75.7172 - val_loss:
136.8328
Epoch 17/100
11392/11392 [==============================] - 4s 309us/step - loss: 75.4796 - val_loss:
136.6859
Epoch 18/100
11392/11392 [==============================] - 3s 305us/step - loss: 75.6170 - val_loss:
135.7118
Epoch 19/100
11392/11392 [==============================] - 4s 308us/step - loss: 75.5432 - val_loss:
136.9410
Epoch 20/100
11392/11392 [==============================] - 3s 298us/step - loss: 75.3167 - val_loss:
137.5757
Epoch 21/100
11392/11392 [==============================] - 3s 301us/step - loss: 75.4451 - val_loss:
136.3058
Epoch 22/100
11392/11392 [==============================] - 3s 302us/step - loss: 75.3998 - val_loss:
137.2671
Epoch 23/100
11392/11392 [==============================] - 3s 298us/step - loss: 75.2278 - val_loss:
136.3566
Epoch 24/100
11392/11392 [==============================] - 3s 295us/step - loss: 74.9034 - val_loss:
135.8716
Epoch 25/100
11392/11392 [==============================] - 3s 293us/step - loss: 74.8696 - val_loss:
135.8586
Epoch 26/100
11392/11392 [==============================] - 3s 292us/step - loss: 74.8821 - val_loss:
135.8913
Epoch 27/100
11392/11392 [==============================] - 3s 289us/step - loss: 74.4589 - val_loss:
135.8890
Epoch 28/100
```

```
11392/11392 [==============================] - 3s 288us/step - loss: 75.0098 - val_loss:
136.7601
Epoch 29/100
11392/11392 [==============================] - 3s 287us/step - loss: 74.8737 - val_loss:
136.4143
Epoch 30/100
11392/11392 [==============================] - 3s 287us/step - loss: 74.8817 - val_loss:
135.9963
Epoch 31/100
11392/11392 [==============================] - 3s 285us/step - loss: 75.0249 - val_loss:
136.2620
Epoch 32/100
11392/11392 [==============================] - 3s 286us/step - loss: 74.8326 - val_loss:
136.2540
Epoch 33/100
11392/11392 [==============================] - 3s 286us/step - loss: 75.0012 - val_loss:
136.3354
Epoch 34/100
11392/11392 [==============================] - 3s 289us/step - loss: 74.9884 - val_loss:
135.7882
Epoch 35/100
11392/11392 [==============================] - 3s 288us/step - loss: 74.9038 - val_loss:
135.1051
Epoch 36/100
11392/11392 [==============================] - 3s 283us/step - loss: 74.4046 - val_loss:
135.9847
Epoch 37/100
11392/11392 [==============================] - 3s 292us/step - loss: 74.5560 - val_loss:
135.9517
Epoch 38/100
11392/11392 [==============================] - 3s 302us/step - loss: 74.6539 - val_loss:
135.8237
Epoch 39/100
11392/11392 [==============================] - 3s 301us/step - loss: 74.3419 - val_loss:
135.6312
Epoch 40/100
11392/11392 [==============================] - 3s 293us/step - loss: 74.0559 - val_loss:
135.6073
Epoch 41/100
11392/11392 [==============================] - 3s 281us/step - loss: 73.9340 - val_loss:
135.1617
Epoch 42/100
11392/11392 [==============================] - 3s 289us/step - loss: 74.3479 - val_loss:
136.0021
Epoch 43/100
11392/11392 [==============================] - 3s 291us/step - loss: 74.2855 - val_loss:
135.6383
Epoch 44/100
11392/11392 [==============================] - 4s 328us/step - loss: 74.0872 - val_loss:
135.6030
Epoch 45/100
11392/11392 [==============================] - 4s 318us/step - loss: 73.7212 - val_loss:
```

```
135.5992
Epoch 46/100
11392/11392 [==============================] - 4s 317us/step - loss: 74.0939 - val_loss:
135.7620
Epoch 47/100
11392/11392 [==============================] - 4s 322us/step - loss: 74.2365 - val_loss:
135.6826
Epoch 48/100
11392/11392 [==============================] - 4s 329us/step - loss: 74.1362 - val_loss:
136.5657
Epoch 49/100
11392/11392 [==============================] - 4s 329us/step - loss: 73.9892 - val_loss:
135.1308
Epoch 50/100
11392/11392 [==============================] - 4s 361us/step - loss: 74.1460 - val_loss:
135.1062
Epoch 51/100
11392/11392 [==============================] - 4s 378us/step - loss: 73.6903 - val_loss:
135.8304
Epoch 52/100
11392/11392 [==============================] - 4s 363us/step - loss: 73.5888 - val_loss:
135.2697
Epoch 53/100
11392/11392 [==============================] - 4s 334us/step - loss: 74.1868 - val_loss:
135.2799
Epoch 54/100
11392/11392 [==============================] - 4s 358us/step - loss: 73.8565 - val_loss:
135.5638
Epoch 55/100
11392/11392 [==============================] - 4s 338us/step - loss: 74.0546 - val_loss:
135.9823
Epoch 56/100
11392/11392 [==============================] - 4s 324us/step - loss: 74.0429 - val_loss:
135.1621
Epoch 57/100
11392/11392 [==============================] - 4s 325us/step - loss: 73.9076 - val_loss:
134.9111
Epoch 58/100
11392/11392 [==============================] - 4s 327us/step - loss: 73.7091 - val_loss:
134.9617
Epoch 59/100
11392/11392 [==============================] - 4s 324us/step - loss: 74.1068 - val_loss:
135.2714
Epoch 60/100
11392/11392 [==============================] - 4s 329us/step - loss: 73.5627 - val_loss:
135.0033
Epoch 61/100
11392/11392 [==============================] - 4s 326us/step - loss: 74.0660 - val_loss:
134.8258
Epoch 62/100
11392/11392 [==============================] - 4s 326us/step - loss: 73.7481 - val_loss:
135.8892
```

```
Epoch 63/100
11392/11392 [==============================] - 4s 315us/step - loss: 73.5941 - val_loss:
136.2032
Epoch 64/100
11392/11392 [==============================] - 4s 314us/step - loss: 73.6514 - val_loss:
136.1213
Epoch 65/100
11392/11392 [==============================] - 4s 308us/step - loss: 73.8684 - val_loss:
135.8529
Epoch 66/100
11392/11392 [==============================] - 3s 303us/step - loss: 73.3982 - val_loss:
135.1600
Epoch 67/100
11392/11392 [==============================] - 5s 415us/step - loss: 73.7812 - val_loss:
135.8742
Epoch 68/100
11392/11392 [==============================] - 5s 420us/step - loss: 73.8326 - val_loss:
136.9918
Epoch 69/100
11392/11392 [==============================] - 5s 430us/step - loss: 73.4467 - val_loss:
134.9832
Epoch 70/100
11392/11392 [==============================] - 5s 414us/step - loss: 73.2215 - val_loss:
135.5477
Epoch 71/100
11392/11392 [==============================] - 5s 411us/step - loss: 73.5330 - val_loss:
134.8381
Epoch 72/100
11392/11392 [==============================] - 5s 410us/step - loss: 73.8583 - val_loss:
135.6827
Epoch 73/100
11392/11392 [==============================] - 5s 402us/step - loss: 74.1662 - val_loss:
135.8720
Epoch 74/100
11392/11392 [==============================] - 5s 401us/step - loss: 73.3522 - val_loss:
135.2027
Epoch 75/100
11392/11392 [==============================] - 5s 397us/step - loss: 73.7884 - val_loss:
135.4805
Epoch 76/100
11392/11392 [==============================] - 5s 406us/step - loss: 73.0115 - val_loss:
135.6366
Epoch 77/100
11392/11392 [==============================] - 5s 402us/step - loss: 73.5386 - val_loss:
135.4864
Epoch 78/100
11392/11392 [==============================] - 5s 406us/step - loss: 73.8190 - val_loss:
136.0704
Epoch 79/100
11392/11392 [==============================] - 5s 405us/step - loss: 73.2153 - val_loss:
135.2263
Epoch 80/100
```

```
11392/11392 [==============================] - 5s 412us/step - loss: 73.2077 - val_loss:
135.9256
Epoch 81/100
11392/11392 [==============================] - 5s 429us/step - loss: 73.5141 - val_loss:
135.2106
Epoch 82/100
11392/11392 [==============================] - 5s 431us/step - loss: 73.7084 - val_loss:
135.5198
Epoch 83/100
11392/11392 [==============================] - 5s 426us/step - loss: 73.7806 - val_loss:
135.7354
Epoch 84/100
11392/11392 [==============================] - 5s 426us/step - loss: 72.9402 - val_loss:
134.7090
Epoch 85/100
11392/11392 [==============================] - 5s 413us/step - loss: 73.1296 - val_loss:
135.2897
Epoch 86/100
11392/11392 [==============================] - 5s 407us/step - loss: 72.5914 - val_loss:
134.8225
Epoch 87/100
11392/11392 [==============================] - 5s 413us/step - loss: 73.3068 - val_loss:
135.0381
Epoch 88/100
11392/11392 [==============================] - 5s 408us/step - loss: 73.6438 - val_loss:
134.9860
Epoch 89/100
11392/11392 [==============================] - 5s 416us/step - loss: 73.1057 - val_loss:
135.7923
Epoch 90/100
11392/11392 [==============================] - 5s 420us/step - loss: 72.6194 - val_loss:
135.1494
Epoch 91/100
11392/11392 [==============================] - 5s 428us/step - loss: 73.3131 - val_loss:
136.1112
Epoch 92/100
11392/11392 [==============================] - 5s 438us/step - loss: 73.2137 - val_loss:
135.5893
Epoch 93/100
11392/11392 [==============================] - 5s 415us/step - loss: 73.2216 - val_loss:
135.2976
Epoch 94/100
11392/11392 [==============================] - 5s 426us/step - loss: 73.1577 - val_loss:
135.5076
Epoch 95/100
11392/11392 [==============================] - 5s 460us/step - loss: 72.9172 - val_loss:
135.3029
Epoch 96/100
11392/11392 [==============================] - 5s 423us/step - loss: 73.4844 - val_loss:
135.6762
Epoch 97/100
11392/11392 [==============================] - 5s 416us/step - loss: 73.0642 - val_loss:
```

```
134.9428
Epoch 98/100
11392/11392 [==============================] - 5s 424us/step - loss: 73.1254 - val_loss:
135.5939
Epoch 99/100
11392/11392 [==============================] - 5s 413us/step - loss: 72.8956 - val_loss:
135.2971
Epoch 100/100
11392/11392 [==============================] - 5s 415us/step - loss: 73.9355 - val_loss:
136.2061
```

Out [78]:

```
<keras.callbacks.History at 0x1a3e70ceb8>
```

In [79]:

```
history_dict = model.history.history
history_dict
```

Out [79]:

```
{'val_loss': [135.16367218805397,
  134.9498313147089,
  135.9605003543522,
  135.47224897405377,
  136.15934493489888,
  138.2725725588591,
  137.4397633827251,
  137.2840695122014,
  136.73512259773585,
  136.38118195792904,
  137.79151871411696,
  137.19939041915148,
  136.23730771697086,
  135.69791185337564,
  137.3448102577873,
  136.83279699346295,
  136.6858893270078,
  135.71175315069115,
  136.94101787650067,
  137.57568349786428,
  136.3057704490164,
  137.26707619687787,
  136.35661663438964,
  135.87159798456275,
  135.8586028399675,
  135.89134248443273,
  135.88904394015023,
  136.76006380630577,
  136.41429796426192,
  135.99628908737847,
  136.26199219019517,
  136.25398697542107,
```

```
136.3354464562043,
135.7881511501644,
135.10505639988443,
135.98467486060184,
135.95169731326726,
135.82371818760166,
135.63121779327807,
135.6072580581126,
135.16173247669053,
136.00214372510496,
135.63827047140703,
135.60304602332738,
135.59922474881878,
135.76196748277417,
135.68255125439686,
136.56572292421174,
135.13083099282306,
135.1062146710313,
135.83044238712478,
135.2697183619375,
135.27985302261683,
135.56384599727133,
135.98227407103,
135.16211633837742,
134.9110767815424,
134.96165508809298,
135.27139737813368,
135.00325930636862,
134.82581366663393,
135.88922186001489,
136.20316625418872,
136.12128876603168,
135.85287400432256,
135.1599527778833,
135.87423627013746,
136.9917834219725,
134.9831859236178,
135.54773557704428,
134.8381004229836,
135.68271297475567,
135.87196521655372,
135.20269951613054,
135.48049013510993,
135.63659933338994,
135.4864233136177,
136.07036948722342,
135.2263261027958,
135.92557903735533,
135.21062144507533,
135.51978121633115,
135.73542984153914,
134.70904507066894,
```

135.28967294485673,
134.82248388684314,
135.0381415512251,
134.985987233079,
135.79231433246446,
135.14935334350753,
136.11117382671523,
135.58925944307575,
135.2975594997406,
135.5076061539028,
135.30286930436674,
135.67622696835062,
134.94275607233462,
135.59392724866453,
135.29707176011541,
136.20606830327407],
'loss': [80.52873749679394,
78.85931565788354,
78.31079526429765,
77.55531261744125,
77.10103403584341,
77.23155845684951,
76.92437426963549,
77.05955593237717,
76.64314084642389,
76.32728873477893,
76.30230967918139,
75.91065584139878,
75.92032957612798,
76.10650432779548,
75.85147920083463,
75.71724877732524,
75.47955340481876,
75.61704067701704,
75.54319012566899,
75.31671532620204,
75.44511173012552,
75.39982192435961,
75.22782209482085,
74.90344038974034,
74.86963635883974,
74.88212640633743,
74.45889989445719,
75.00979293330332,
74.87369550747817,
74.88174357574977,
75.0249239112554,
74.83261706587973,
75.00120898311057,
74.98841368482354,
74.90376163868422,
74.40458837519871,

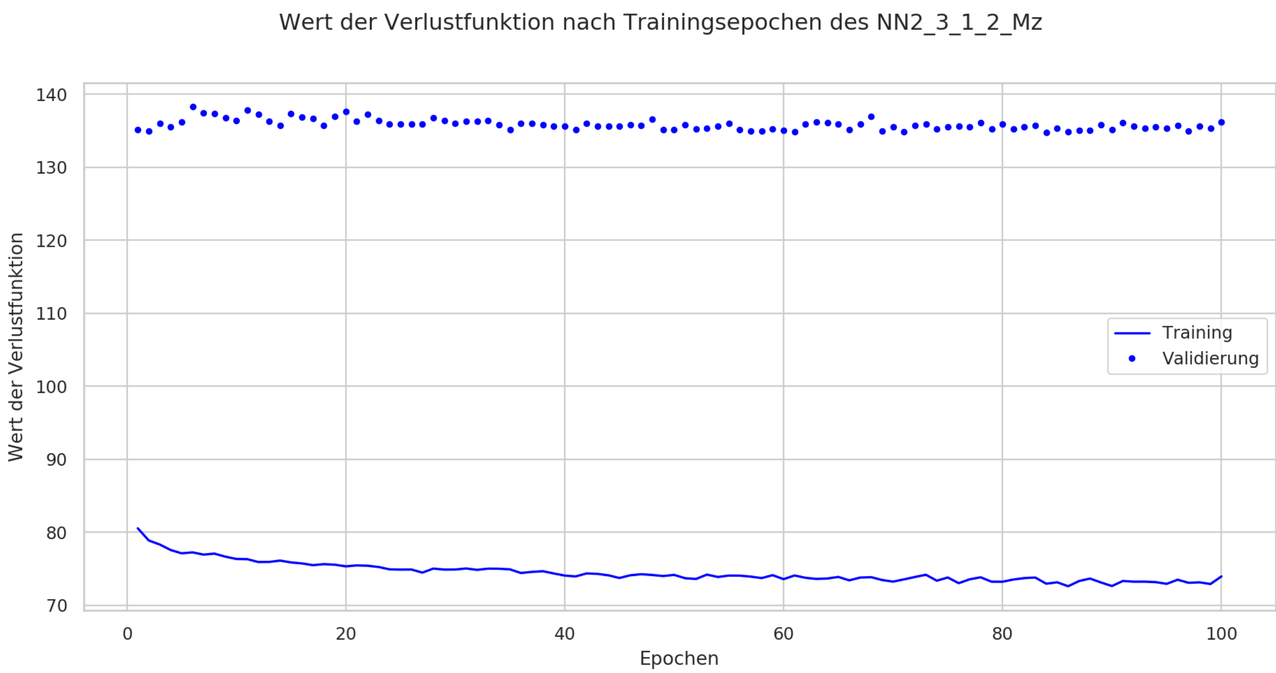
```
74.55597837855306,
74.65385793032272,
74.34193782324202,
74.05585404996121,
73.93397143985449,
74.34794874405593,
74.28552608811454,
74.08716332510616,
73.72123415014717,
74.09393926684776,
74.23654380005397,
74.1361837119199,
73.98915222253692,
74.14603814114345,
73.69030045123583,
73.58884680672978,
74.18681363309368,
73.85646816317954,
74.05464876903577,
74.04294901215628,
73.90755099660895,
73.70909889360492,
74.10682381672805,
73.56270437562064,
74.06601616505826,
73.74805151210742,
73.59408312701107,
73.65137224250965,
73.86835699938656,
73.39817917748783,
73.78115742929866,
73.83259058802315,
73.44666575313954,
73.2214917574036,
73.5330183639955,
73.85832066214486,
74.16616173808494,
73.35217897275861,
73.78838314367144,
73.01148851533954,
73.53858940253097,
73.81896630297886,
73.21533592095535,
73.20770097582528,
73.51410372337598,
73.70840696806319,
73.7805835530999,
72.94024623645825,
73.12955344125126,
72.59139909369222,
73.30682503507379,
73.643768519498,
```

```
73.10568750842233,
73.61937303489513,
73.31310779592964,
73.2136955636271,
73.22164767779661,
73.15770812516801,
72.91721688227707,
73.48444824004442,
73.06423841969351,
73.1253980304418,
72.89557385980413,
73.9355085887266]}
```

Analysiere Trainingsergebnisse

In [80]:

```
sns.set_context("paper")

loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values)+1)

plt.figure(num=None, figsize=(11,5), dpi=200, facecolor='w', edgecolor='k')
#f, (ax1,ax2) = plt.subplots(1,2, figsize=(11, 5), dpi=100, facecolor='w', edgecolor='k')
#f.suptitle('Trainingsverlauf '+name)
plt.plot(epochs, loss_values, 'b',label='Training')
plt.plot(epochs, val_loss_values, 'b.',label='Validierung')
plt.suptitle('Wert der Verlustfunktion nach Trainingsepochen des '+name)
plt.xlabel('Epochen')
plt.ylabel('Wert der Verlustfunktion')

plt.legend()#bbox_to_anchor=(0.9, 0., 0.5, 0.5), borderaxespad=1)
plt.show()
```



Anwendung des trainierten Models auf 'unbekannte' Trainingsdaten

In [6]:

```
predictions = model.predict(x_test,batch_size=batchsize)
y_real = y_test
```

In [82]:

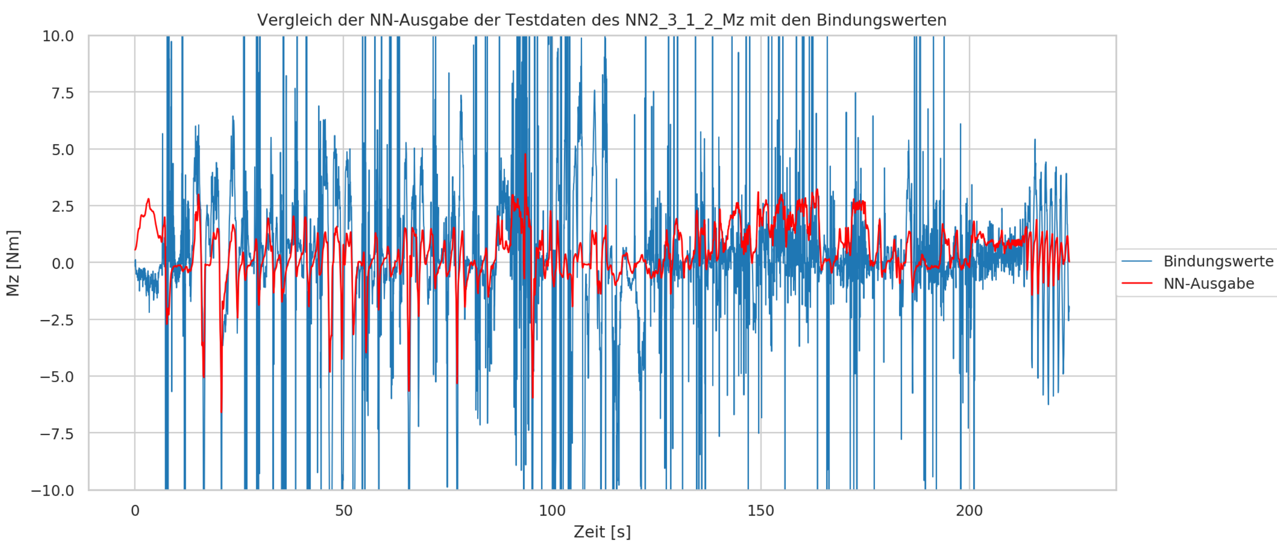
```
predictions.shape
```

Out [82]:

```
(4480, 1)
```

In [83]:

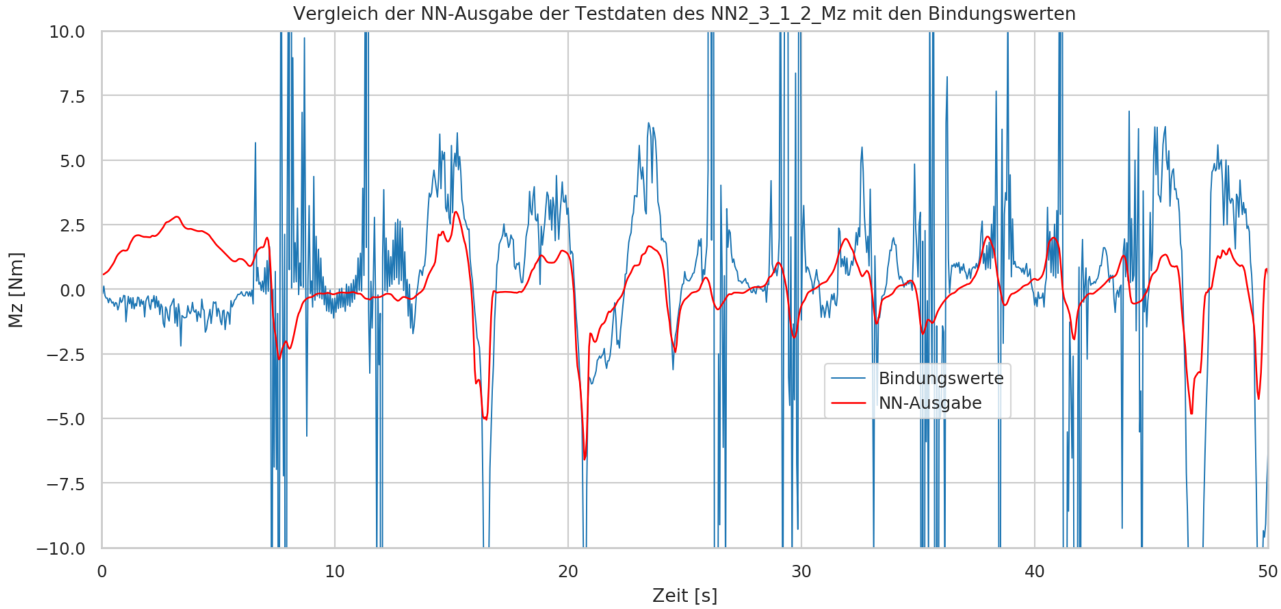
```
plt.figure(num=None, figsize=(11,5), dpi=200, facecolor='w', edgecolor='k')
plt.plot(np.linspace(0,len(predictions)*0.05-0.05,len(predictions)), y_real, label='Bindungswerte',
plt.plot(np.linspace(0,len(predictions)*0.05-0.05,len(predictions)), predictions,'r', label='NN-Ausg

plt.title('Vergleich der NN-Ausgabe der Testdaten des ' +name +' mit den Bindungswerten')
plt.xlabel('Zeit [s]')
plt.ylabel('Mz [Nm]')
plt.legend(bbox_to_anchor=(0.61, 0.15, 0.55, 0.38), borderaxespad=0)
#plt.xlim(left=140, right=220)
#plt.xlim(left=150, right=200)
plt.ylim(bottom=-10, top=10)
plt.show()
```



In [84]:

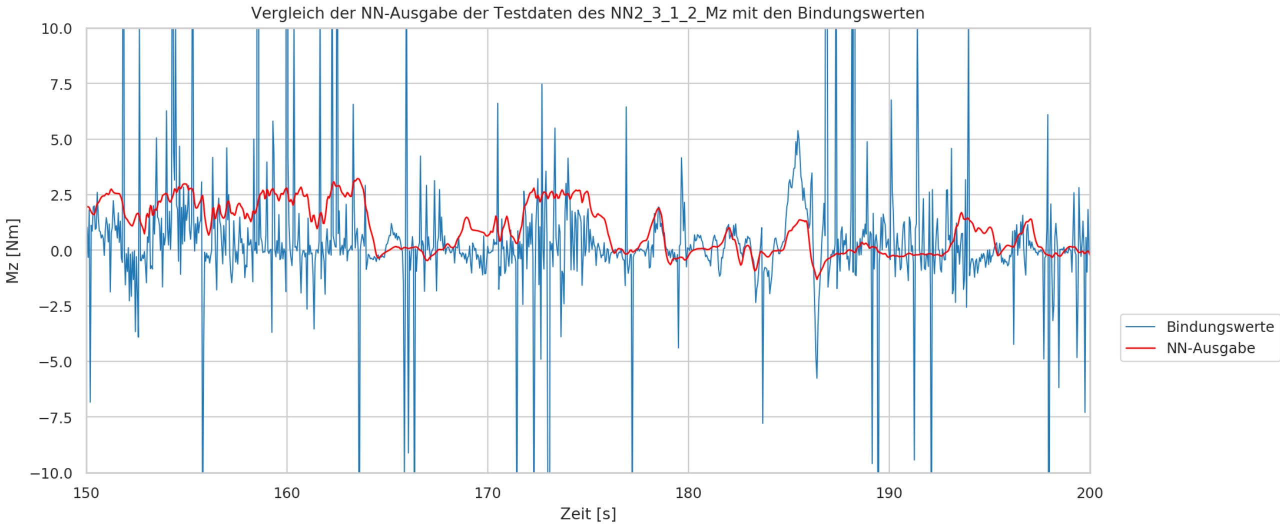
```
plt.figure(num=None, figsize=(11,5), dpi=200, facecolor='w', edgecolor='k')
plt.plot(np.linspace(0,len(predictions)*0.05-0.05,len(predictions)), y_real, label='Bindungswerte',
plt.plot(np.linspace(0,len(predictions)*0.05-0.05,len(predictions)), predictions,'r', label='NN-Ausg
```

```
plt.title('Vergleich der NN-Ausgabe der Testdaten des ' +name +' mit den Bindungswerten')
plt.xlabel('Zeit [s]')
plt.ylabel('Mz [Nm]')
plt.legend(bbox_to_anchor=(0.62, 0.25, 0.58, 0.45), borderaxespad=0)
plt.xlim(left=0, right=50)
#plt.xlim(left=150, right=200)
plt.ylim(bottom=-10, top=10)
plt.show()
```



In [85]:

```
plt.figure(num=None, figsize=(11,5), dpi=200, facecolor='w', edgecolor='k')
plt.plot(np.linspace(0,len(predictions)*0.05-0.05,len(predictions)), y_real, label='Bindungswerte',
plt.plot(np.linspace(0,len(predictions)*0.05-0.05,len(predictions)), predictions,'r', label='NN-Ausg

plt.title('Vergleich der NN-Ausgabe der Testdaten des ' +name +' mit den Bindungswerten')
plt.xlabel('Zeit [s]')
plt.ylabel('Mz [Nm]')
plt.legend(bbox_to_anchor=(0.61, 0.25, 0.58, 0.45), borderaxespad=0)
#plt.xlim(left=0, right=50)
plt.xlim(left=150, right=200)
plt.ylim(bottom=-10, top=10)
plt.show()
```



Zusammenfassung

In [86]:

```
# Nach welcher Epoche sollte das Training optimalerweise abgeschlossen werden?
# Vorgehen: einmal Netzwerk berechnen in 25 Epochen -> optimale Epochenanzahl anhand 'min_index_val_
#           danach Netzwerk neu berechnen lassen.
min_index_val_loss, min_value_val_loss = min(enumerate(history_dict['val_loss']), key=operator.itemg
#max_index_val_acc, max_value_val_acc = max(enumerate(history_dict['val_acc']), key=operator.itemget
print('Ergebnisse der Validierungsdaten:')
print('  opimale Epochenanzahl:                    '+str(min_index_val_loss+1))
print('  minimaler Verlust:                        '+str(min_value_val_loss)+'\n')

print('Ergebnisse der Trainingdaten zur optimalen Epochenzahl:')
print('  Verlust:                                  '+str(history_dict['loss'][min_index_val_loss]) + '\
```

```
Ergebnisse der Validierungsdaten:
  opimale Epochenanzahl:                    84
  minimaler Verlust:                        134.70904507066894

Ergebnisse der Trainingdaten zur optimalen Epochenzahl:
  Verlust:                                  72.94024623645825
```

In [87]:

```
# Beurteilung der Testdaten: Vergleich von 'predictions' mit y_real
#                    - Kreuzkorrelation
#                    - Euklidsche Distanz



#   Kreuzkorrelation
print('Vergleich der Vorhersagewerte mit den Bindungswerten:')
print('  Korrelationskoeffizient:                  '+ str(np.corrcoef(np.transpose(predictions),np.tran
```